Wydanie VIII

Rogers Cadenhead

Java

w 24 godziny

SAMS

Helion

Tytuł oryginału: Sams Teach Yourself Java in 24 Hours, Eighth Edition

Tłumaczenie: Zbigniew Waśko (wstęp, rozdz. 1 – 7, 10 – 24, dodatki), Paweł Jóźwiak (rozdz. 8, 9)

ISBN: 978-83-283-4235-4

Authorized translation from the English language edition, entitled: JAVA IN 24 HOURS, SAMS TEACH YOURSELF (COVERING JAVA 9), Eighth Edition; ISBN 0672337940;
by Rogers Cadenhead; published by Pearson Education, Inc, publishing as SAMS Publishing.

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki
Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock Images LLC.

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *http://helion.pl* (księgarnia internetowa, katalog książek)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:
ftp://ftp.helion.pl/przyklady/jav24h.zip

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
http://helion.pl/user/opinie/jav24h
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

Spis treści

Godzina 2.

Napisz swój pierwszy program

Zagadnienia do nauki:

- **wpisywanie programu w edytorze tekstu,**
- **porządkowanie programu za pomocą nawiasów,**
- **zapisywanie informacji w zmiennej,**
- **wyświetlanie informacji przechowywanej w zmiennej,**
- **zapisywanie, kompilowanie i uruchamianie programu.**

Jak już pisałem w godzinie 1., „Zostań programistą", program komputerowy jest zestawem instrukcji, które mówią komputerowi, co ma robić. Instrukcje te są przekazywane komputerowi przy użyciu języka programowania.

W ciągu tej godziny utworzysz swój pierwszy program w Javie, wpisując instrukcje w edytorze tekstu. Następnie zapiszesz program, skompilujesz go i przetestujesz. Potem spróbujesz go celowo zepsuć i naprawić, tylko po to, żeby zobaczyć, jak to się robi.

Co jest potrzebne do napisania programu?

Jak już wiadomo z godziny 1., do tworzenia programów w Javie potrzebne jest narzędzie programistyczne obsługujące pakiet Java Development Kit (JDK), jakim jest np. zintegrowane środowisko programistyczne (IDE) NetBeans. Potrzebujesz narzędzia, które skompiluje i uruchomi programy napisane w Javie, oraz edytor tekstowy, w którym napiszesz te programy.

W większości przypadków programy komputerowe pisze się w edytorze tekstu (zwanym także edytorem kodu źródłowego). Niektóre języki programowania mają własne edytory. Środowisko NetBeans zawiera taki edytor do pisania programów w Javie.

Programy pisane w Javie są zwykłymi plikami tekstowymi bez specjalnego formatowania, np. wyśrodkowania lub pogrubienia. Edytor kodu źródłowego w NetBeans działa jak prosty edytor tekstu, ale jest wyposażony w kilka usprawnień przydatnych dla programistów. Podczas wprowadzania tekstu zmienia się jego kolor zgodnie z poszczególnymi elementami języka. NetBeans tworzy także odpowiednie wcięcia akapitów i zapewnia pomocną dokumentację programistyczną.

Ponieważ programy Javy są plikami tekstowymi, można je otwierać i edytować w każdym edytorze tekstu. Możesz napisać program w środowisku NetBeans, otworzyć go w windowsowym Notatniku, wprowadzić tam zmiany i ponownie, bez żadnych problemów, otworzyć w edytorze NetBeans.

Przygotowanie do pisania programu Saluton

Pierwszy program, który napiszesz, wyświetla tradycyjne w świecie informatyków pozdrowienie: „Saluton mondo!" (czyli „Witaj, świecie").

Jeśli jeszcze tego nie zrobiłeś, przygotuj pierwszy projekt programistyczny w NetBeans. W tym celu utwórz nowy projekt o nazwie *Java24*, wykonując następujące czynności:

1. Z menu *File* wybierz polecenie *New Project*. Otworzy się okno dialogowe *New Project*.
2. Wybierz kategorię projektu *Java* oraz typ projektu *Java Application*, a następnie kliknij przycisk *Next*.
3. Jako nazwę projektu wpisz **`Java24`** (jeśli wcześniej utworzyłeś już projekt o tej nazwie, pojawi się komunikat o błędzie informujący, że taki folder już istnieje i nie jest pusty — *Project folder already exists and is not empty*).
4. Wyłącz tworzenie klasy głównej przez usunięcie zaznaczenia pola wyboru *Create Main Class*.
5. Kliknij przycisk *Finish*, aby zakończyć tworzenie nowego projektu.

Dla projektu Java24 został utworzony nowy folder. Z tego projektu możesz korzystać przy pisaniu wszystkich programów Javy omawianych w tej książce.

Rozpoczynamy pisanie programu

NetBeans grupuje powiązane programy we wspólny projekt. Jeśli projekt Java24 nie jest otwarty, wykonaj poniższe czynności:

1. Wybierz polecenie *File/Open Project*. Pojawi się okno dialogowe wyboru pliku.
2. Odszukaj i wskaż folder *NetBeansProjects* (jeśli nie jest jeszcze wybrany).
3. Zaznacz pozycję *Java24* i kliknij przycisk *Open Project*.

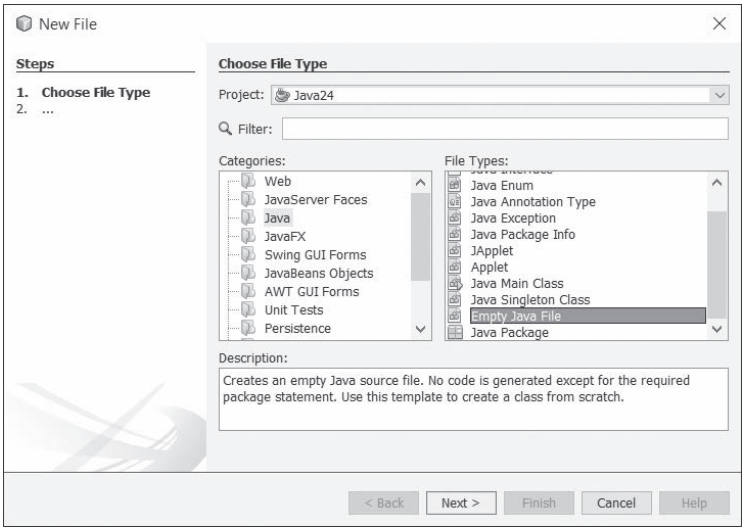
Projekt Java24 pojawi się na panelu *Projects* obok ikony filiżanki kawy oraz znaku plus (*+*), którego kliknięcie rozwija listę plików i folderów wchodzących w skład projektu.

Aby dodać nowy program do aktualnie otwartego projektu, wybierz polecenie *File/New File*. Otworzy się kreator *New File* pokazany na rysunku 2.1.

W panelu *Categories* wyświetlane są różne rodzaje programów, które możesz utworzyć. Kliknij pozycję *Java*, aby zobaczyć, jakie typy plików należą do tej kategorii. Na potrzeby swojego pierwszego projektu wybierz typ *Empty Java File* (na końcu listy *File Types*) i kliknij przycisk *Next*.

Otworzy się okno dialogowe *New Empty Java File*. Wykonaj poniższe czynności, aby rozpocząć pisanie programu:

1. W polu tekstowym *Class Name* wpisz nazwę klasy **`Saluton`**.
2. W polu *Package* wpisz nazwę pakietu **`com.java24hours`**.
3. Kliknij przycisk *Finish*.



RYSUNEK 2.1. Kreator nowego pliku

Możesz już teraz rozpocząć pracę nad swoim programem — w edytorze kodu źródłowego otworzy się pusty plik o nazwie *Saluton.java*. Korzystając z tego edytora, rozpocznij karierę programisty i wprowadź wszystkie wiersze z listingu 2.1. Zawarte tam instrukcje noszą nazwę kodu źródłowego programu.

LISTING 2.1. Program Saluton

```
1: package com.java24hours;
2:
3: class Saluton {
4:    public static void main(String[] arguments) {
5:       // Tutaj trafi mój pierwszy program w Javie.
6:    }
7: }
```

Ostrzeżenie

> Nie wprowadzaj numeru wiersza i dwukropka na początku poszczególnych wierszy — są one dodawane jedynie po to, aby ułatwić odnoszenie się do poszczególnych fragmentów kodu.

Pamiętaj, aby przepisać wszystko dokładnie tak, jak jest w listingu (włącznie z wielkością liter), i za pomocą spacji lub klawisza *Tab* wprowadź wcięcia w wierszach od 4. do 6. Gdy skończysz, wybierz polecenie *File/Save*, aby zapisać plik.

W tym momencie plik *Saluton.java* zawiera jedynie podstawy programu Java. Napiszesz wiele programów, które będą wyglądać tak jak ten, z wyjątkiem słowa `Saluton` w wierszu 3.

Stanowi ono nazwę programu i za każdym razem będzie inne. Zawartość wiersza 5. na pewno rozumiesz, ponieważ jest napisany w języku, którego używasz na co dzień. Reszta może być nowością.

Instrukcja class

Pierwszy wiersz programu na następującą postać:

```
package com.java24hours;
```

Pakiet (`package`) to w Javie sposób na grupowanie programów. Ten wiersz informuje komputer, aby nadał pakietowi nazwę `com.java24hours`.

Po pustym wierszu znajduje się trzeci wiersz o treści:

```
class Saluton {
```

W przekładzie na język codzienny oznacza to: „Komputerze, nadaj mojemu programowi nazwę Saluton".

Jak pewnie przypominasz sobie z godziny 1., każde polecenie wydawane komputerowi jest nazywane instrukcją. Za pomocą instrukcji `class` nadajesz programowi komputerowemu nazwę. Służy ono także do zdefiniowania innych danych programu, o których dowiesz się nieco później. Pojęcie klasy (`class`) ma istotne znaczenie, ponieważ programy pisane w Javie także są nazywane klasami.

W prezentowanym przykładzie nazwa programu Saluton odpowiada nazwie pliku *Saluton.java*. Program w Javie musi mieć nazwę pasującą do pierwszej części nazwy jego pliku, czyli tej, która znajduje się przed znakiem kropki (`.`). Obie nazwy powinny być także napisane literami o takiej samej wielkości.

Jeśli nazwa programu nie odpowiada nazwie pliku, próba skompilowania niektórych programów może zakończyć się komunikatem o błędzie — wszystko zależy od tego, w jaki sposób instrukcja `class` została użyta do skonfigurowania programu.

Za co odpowiada instrukcja main?

Kolejny wiersz programu ma następującą postać:

```
public static void main(String[] arguments) {
```

Mówi on komputerowi: „Tutaj zaczyna się główna część programu". Programy w Javie składają się z różnych sekcji. Musi więc istnieć sposób na zidentyfikowanie tej części programu, która po jego uruchomieniu jest wykonywana jako pierwsza.

Instrukcja `main` jest punktem początkowym większości programów pisanych w Javie. Wyjątki stanowią aplety, czyli programy uruchamiane na stronie internetowej przez przeglądarkę, serwlety, czyli programy uruchamiane przez serwer sieciowy, oraz aplikacje mobilne, czyli programy pracujące na urządzeniach mobilnych.

Większość programów opisanych w tej książce używa instrukcji `main` jako punktu początkowego. Wynika to z faktu, że będą uruchamiane na komputerze. Aplety i serwlety są uruchamiane pośrednio przez inny program lub urządzenie.

Dla odróżnienia programy uruchamiane bezpośrednio przez użytkownika są nazywane aplikacjami.

Nawiasy klamrowe

W programie Saluton wiersze 3., 4., 6. i 7. zawierają znaki nawiasów klamrowych { oraz }. Służą one do grupowania wierszy programu (podobnie jak zwykłe nawiasy grupują słowa w zdaniu). Wszystko, co znajduje się między nawiasem otwierającym ({) a zamykającym (}), należy do tej samej grupy.

Takie zgrupowania noszą nazwę bloków. W listingu 2.1 nawias otwierający w wierszu 3. jest związany z nawiasem zamykającym w wierszu 7., tworząc w ten sposób blok stanowiący cały program. Takie stosowanie nawiasów pozwala wskazać początek i koniec programu.

Bloki mogą znajdować się także wewnątrz innych bloków (podobnie jak nawiasy stosowane w tym zdaniu (w którym użyty jest też drugi zestaw nawiasów)). W wierszach 4. i 6. programu Saluton znajdują się nawiasy definiujące kolejny blok. Blok ten zaczyna się od instrukcji `main`. Wiersze należące do bloku `main` będą wykonywane jako pierwsze po uruchomieniu programu.

Wskazówka

> NetBeans pomaga określić miejsca, w których dany blok się rozpoczyna i kończy. Kliknij jeden z nawiasów klamrowych w kodzie źródłowym programu Saluton. Nawias ten podświetli się na żółto wraz z odpowiadającym mu drugim nawiasem. Instrukcje Javy zamknięte między tymi dwoma nawiasami tworzą blok. Ta wskazówka nie jest szczególnie przydatna w przypadku tak krótkiego programu jak Saluton, ale przy pisaniu dłuższych kodów znacznie ułatwi Ci pracę.

Wewnątrz tego bloku znajduje się jedynie poniższa informacja:

```
// Tutaj trafi mój pierwszy program w Javie.
```

Wiersz ten jest tzw. wypełniaczem. Dwa znaki ukośnika (//) na początku wiersza mówią komputerowi, aby go zignorował, ponieważ wiersz został umieszczony jedynie dla osób analizujących kod źródłowy. Wiersze służące do takiego celu określa się jako *komentarze*.

Tak oto napisałeś kompletny program w Javie. Możesz go skompilować, ale po uruchomieniu nic się nie stanie. Nie powiedziałeś bowiem jeszcze komputerowi, co ma robić. Blok instrukcji `main` zawiera jedynie komentarz, który jest ignorowany przez komputer. Musisz wstawić jakieś instrukcje między nawiasami otwierającym i zamykającym bloku `main`.

Przechowywanie informacji w zmiennych

W swoich programach będziesz potrzebował miejsca, w którym byłoby można przechować informację przez pewien czas. Służy do tego zmienna, czyli przestrzeń magazynowa przechowująca informacje, takie jak liczby całkowite, liczby zmiennoprzecinkowe, wartości prawda/fałsz, znaki i łańcuchy tekstowe. Informacja zapisana w zmiennej może się zmieniać — i właśnie dlatego ten element nosi nazwę zmiennej.

W pliku *Saluton.java* zastąp wiersz 5. następującym:

```
String greeting = "Saluton mondo!";
```

Ta instrukcja mówi komputerowi, aby zapisał tekst „Saluton mondo!” w zmiennej `greeting`.

W programie pisanym przy użyciu Javy musisz powiedzieć komputerowi, jaki rodzaj informacji ma być przechowywany w zmiennej. W tym programie zmienna `greeting` jest typu łańcuchowego — może przechowywać ciągi znaków, takich jak litery, cyfry czy znaki interpunkcyjne. Wprowadzone do instrukcji słowo `String` definiuje zmienną jako zdolną do przechowywania łańcuchów.

Przy wprowadzaniu tej instrukcji do programu należy ją zakończyć średnikiem. Średniki kończą wszystkie instrukcje programów pisanych w Javie. Są czymś w rodzaju kropki na końcu zdania. Na ich podstawie komputer może określić, gdzie kończy się jedna instrukcja i rozpoczyna następna.

Wprowadzanie każdej instrukcji w odrębnym wierszu sprawia, że program jest bardziej zrozumiały (dla nas, ludzi).

Wyświetlanie zawartości zmiennej

Jeśli teraz uruchomisz program, nadal będzie się wydawało, że nic się nie dzieje. Polecenie zapisania tekstu w zmiennej `greeting` działa niejako w tle. Jeśli chcesz się przekonać, że komputer jednak coś zrobił, możesz wyświetlić zawartość zmiennej.

Wstaw kolejny pusty wiersz w programie Saluton, pod instrukcją `String greeting = "Saluton mondo!"`. W nowym wierszu wpisz następującą instrukcję:

```
System.out.println(greeting);
```

Ta instrukcja mówi komputerowi, aby wyświetlił wartość przechowywaną w zmiennej `greeting`. Instrukcja `System.out.println` każe komputerowi wyświetlić informację na systemowym urządzeniu wyjściowym — na monitorze.

To jest już coś.

Zapisywanie kompletnego programu

Twój program powinien teraz przypominać ten z listingu 2.2, chociaż może mieć nieco inne odstępy w wierszach 5. i 6. Wykonaj potrzebne poprawki i zapisz plik (za pomocą polecenia *File/Save*).

LISTING 2.2. Gotowa wersja programu Saluton

```
1: package com.java24hours;
2:
3: class Saluton {
4:    public static void main(String[] arguments) {
5:       String greeting = "Saluton mondo!";
6:       System.out.println(greeting);
7:    }
8: }
```

Gdy komputer uruchomi ten program, wykona poszczególne instrukcje w ramach bloku `main` umieszczone w wierszach 5. i 6. W listingu 2.3 pokazano, jak wyglądałby program, gdyby został napisany w języku polskim, a nie w Javie.

LISTING 2.3. Opis poszczególnych wierszy programu Saluton

```
1: Umieść ten program w pakiecie com.java24hours.
2:
3: Tutaj rozpoczyna się program Saluton:
4:    Tutaj rozpoczyna się główna część programu:
5:       Zapisz tekst „Saluton mondo!" w zmiennej łańcuchowej o nazwie greeting.
6:       Wyświetl zawartość zmiennej greeting.
7:    Tutaj kończy się główna część programu.
8: Tutaj kończy się program Saluton.
```

Listing 2.4 pokazuje, jak wyglądałby ten program, gdyby był napisany w języku klingońskim, czyli języku wojowników z serialu *Star Trek*.

LISTING 2.4. Program Saluton w języku klingońskim

```
1: Ten program należy do rodu com.java2hours!
2:
3: Jeśli nie chcesz, by spotkało Cię coś złego, rozpocznij teraz program Saluton!
4:    Tutaj uroczyście rozpoczyna się główna część programu!
5:       Zapisz bełkot „Saluton mondo!" w zmiennej łańcuchowej o nazwie greeting.
6:       Wyświetl ten bełkot w języku klingońskim!
7:    Aby nie narazić się na mój gniew, zakończ główną część programu!
8: Zakończ teraz program Saluton i ciesz się, że uszedłeś z życiem!
```

Kompilowanie programu do pliku klasy

Zanim uruchomisz program napisany w Javie, musisz go skompilować. Podczas kompilacji instrukcje wydawane w ramach programu są przekształcane do postaci zrozumiałej dla komputera.

NetBeans kompiluje programy automatycznie podczas ich zapisywania. Jeśli udało Ci się wprowadzić wszystko tak jak na listingu 2.2, program skompiluje się prawidłowo.

Zostanie utworzona skompilowana wersja programu, czyli nowy plik o nazwie *Saluton.class*. Wszystkie programy pisane w Javie są kompilowane do plików klasy, które otrzymują rozszerzenie *.class*. Program w Javie może się składać z wielu współpracujących ze sobą klas, ale w tak prostym programie jak Saluton potrzebna jest tylko jedna.

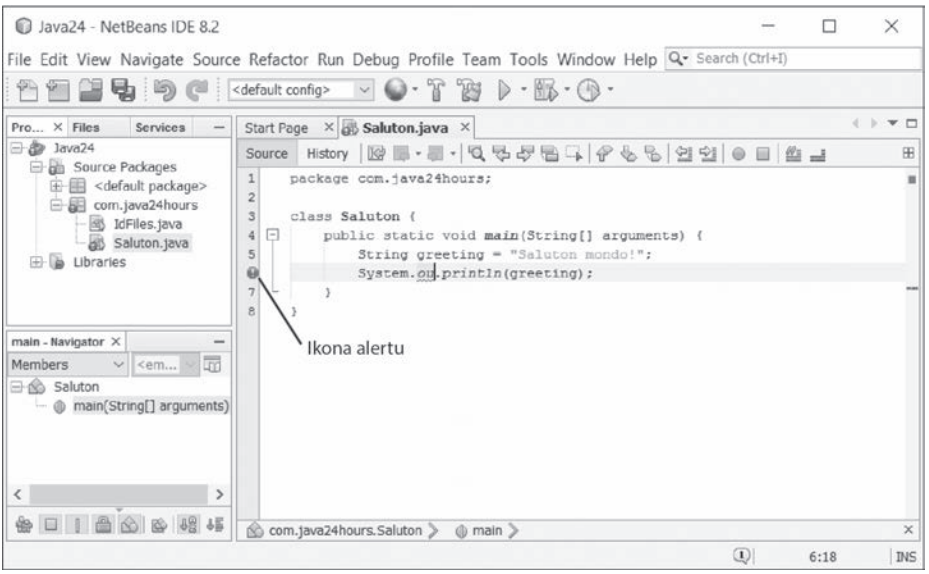
Kompilator przekształca kod źródłowy Javy na kod bajtowy, czyli nadaje mu postać, która może być wykonywana przez wirtualną maszynę Javy (JVM).

Uwaga

> Kompilator Javy odzywa się tylko wtedy, gdy wykryje jakiś błąd, o którym musi poinformować. Jeśli kompilacja przebiegnie pomyślnie bez żadnych problemów, nie wygeneruje żadnej odpowiedzi.
>
> Jest to trochę frustrujące. Kiedy zaczynałem swoją przygodę z Javą, miałem nadzieję, że udanej kompilacji będą towarzyszyć fanfary.

Naprawianie błędów

Podczas pisania programu w edytorze kodu źródłowego, będącym częścią środowiska NetBeans, błędy są oznaczane czerwoną ikoną alertu wyświetlaną przy lewej krawędzi panelu edytora, jak pokazano na rysunku 2.2.



RYSUNEK 2.2. Sygnalizacja błędów w edytorze kodu źródłowego

Ikona wyświetla się w wierszu, w którym błąd został wykryty. Możesz ją kliknąć, aby wyświetlić okno dialogowe błędu zawierające następujące informacje:

- nazwa programu w Javie,
- typ błędu,
- wiersz, w którym wykryto błąd.

Oto przykład zawartości okna błędu, które może się wyświetlić przy kompilowaniu programu Saluton:

```
cannot find symbol.
symbol  : variable greting
location: class Saluton
```

Komunikatem o błędzie jest pierwszy wiersz w oknie: `cannot find symbol` (nie można odnaleźć symbolu). Te komunikaty często mogą być mylące dla początkujących programistów. Jeśli komunikat o błędzie nie ma według Ciebie sensu, nie trać zbyt wiele czasu, aby go rozszyfrować. Lepiej przyjrzeć się problematycznemu wierszowi i sprawdzić go pod kątem najbardziej oczywistych przyczyn.

Przykładowo, czy potrafisz stwierdzić, co jest nie tak w poniższej instrukcji?

```
System.out.println(greting);
```

Błąd polega na literówce w nazwie zmiennej, która powinna mieć postać `greeting` zamiast `greting`. Wprowadź tę literówkę celowo w edytorze NetBeans i zobacz, co się stanie.

Jeśli podczas pisania programu Saluton pojawią się okna z błędami, ponownie sprawdź, czy kod odpowiada listingowi 2.2 i popraw wykryte nieścisłości. Upewnij się, że wielkość liter jest prawidłowa i że wstawiłeś wszystkie znaki interpunkcyjne, np. {, } oraz `;`.

Bardzo często krótkie spojrzenie na wiersz wskazany w oknie błędu wystarczy do znalezienia błędu (lub błędów) i naprawienia go.

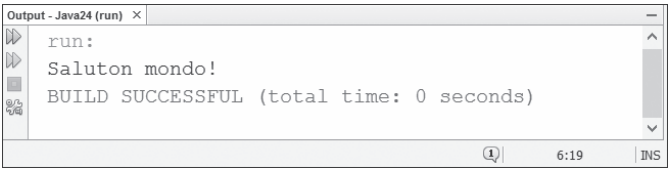
Wskazówka

> Na serwerze FTP wydawnictwa: *ftp://ftp.helion.pl/przyklady/jav24h.zip* znajdują się kody źródłowe wszystkich prezentowanych tu programów.

Uruchamianie programu

Aby sprawdzić, czy program Saluton rzeczywiście wykonuje to, co powinien, uruchom go za pomocą interpretera Javy. W NetBeans wybierz polecenie *Run/Run File*. Poniżej edytora kodu źródłowego otworzy się panel *Output*. Jeśli kod nie zawiera błędów, w tym miejscu wyświetli się rezultat działania programu, jak pokazano na rysunku 2.3.

Jeśli wyświetli się tekst „Saluton mondo!”, to oznacza, że właśnie udało Ci się napisać pierwszy działający program w Javie! Komputer przywitał się ze światem — jest to tradycja w świecie programistów tak samo ważna, jak napoje energetyczne, koszule z krótkimi rękawami i League of Legends.



RYSUNEK 2.3. Uruchamianie pierwszego programu napisanego w Javie

Pewnie zastanawiasz się, dlaczego tradycyjnym pozdrowieniem jest właśnie „Saluton mondo!". Zdanie to oznacza „Witaj, świecie!" w esperanto, sztucznym języku opracowanym przez Ludwika Zamenhofa w 1887 roku w celu ułatwienia komunikacji międzynarodowej.

To pozdrowienie jest tradycyjne tylko dlatego, że właśnie próbuję rozpocząć taką tradycję.

Wskazówka

> Na swoich stronach internetowych firma Oracle udostępnia wyczerpującą dokumentację języka Java. Nie jest Ci ona potrzebna przy czytaniu tej książki, ponieważ każdy temat jest tu dokładnie omawiany, ale jeśli będziesz chcieć poszerzać swoją wiedzę i pisać własne programy, takie zasoby mogą Ci się przydać.
>
> Całą dokumentację można pobrać, ale wygodniejsze jest wyszukiwanie potrzebnych informacji na stronie internetowej. Najbardziej aktualna dokumentacja języka Java znajduje się pod adresem *https://docs.oracle.com/javase/9/*.

Podsumowanie

Podczas tej godziny miałeś szansę napisać swój pierwszy program w Javie. Dowiedziałeś się, że w celu opracowania takiego programu trzeba wykonać cztery następujące czynności podstawowe:

1. Napisać program w edytorze tekstowym lub przy użyciu narzędzia takiego jak NetBeans.
2. Skompilować program do pliku klasy.
3. Rozkazać wirtualnej maszynie Javy, aby uruchomiła klasę.
4. Powiadomić o tym swoją mamę.

Przy okazji poznałeś kilka podstawowych pojęć programistycznych, takich jak kompilatory, interpretery, bloki, instrukcje i zmienne. Zostaną one dokładniej wyjaśnione w trakcie następnych godzin. Jeśli w ciągu tej godziny udało Ci się napisać działający program Saluton, możesz przejść dalej.

(Czwarty krok nie ma nic wspólnego z programowaniem w Javie. Po prostu mama poprosiła mnie, abym o niej wspomniał w książce).

Warsztaty

Pytania i odpowiedzi

P. Jakie znaczenie ma umieszczenie odpowiedniej liczby odstępów na początku wiersza w programie pisanym w Javie?

O. Dla komputera nie ma to żadnego znaczenia. Odstępy są przeznaczone jedynie dla ludzi przeglądających program komputerowy — kompilator Javy ignoruje je zupełnie. Możesz napisać program Saluton bez wcięć wykonywanych przy użyciu klawisza spacji lub *Tab*, a i tak zostanie on pomyślnie skompilowany.

Choć liczba spacji na początku wiersza nie ma znaczenia dla komputera, to jednak warto stosować spójny system odstępów i wcięć w swoich programach. Dlaczego? Ponieważ odpowiednie wcięcia ułatwiają zorientowanie się w strukturze programu oraz szybkie sprawdzenie, do której instrukcji należy dany blok.

Programy, które piszesz, powinny być zrozumiałe dla innych programistów, a także dla Ciebie, gdy po kilku tygodniach lub miesiącach spróbujesz naprawić jakiś błąd lub wprowadzić zmianę. Spójne odstępy i wcięcia są czymś, co nazywany stylem programowania. Dobrzy programiści mają swój styl i go zawsze stosują.

P. Program pisany w języku Java został określony jako klasa i jako grupa klas. Czym jest w rzeczywistości?

O. Jednym i drugim. Proste programy, które napiszesz w Javie w ciągu kilku następnych godzin, są kompilowane do jednego pliku z rozszerzeniem *.class*. Możesz je uruchomić za pomocą wirtualnej maszyny Javy. Programy napisane w Javie mogą jednak składać się także z wielu współpracujących ze sobą klas. Ten temat został szczegółowo opisany w godzinie 10. „Utwórz swój pierwszy obiekt”.

P. Skoro każda instrukcja musi być zakończona średnikiem, to dlaczego wiersz komentarza `// Tutaj trafi mój pierwszy program w Javie` nie kończy się w taki sposób?

O. Komentarze są całkowicie ignorowane przez kompilator. Jeśli wstawisz znaki // do wiersza programu, kompilator zignoruje wszystko to, co znajduje się na prawo od tych znaków. W poniższym przykładzie pokazano komentarz umieszczony w tym samym wierszu co instrukcja:

```
System.out.println(greeting); // Witaj, świecie!
```

P. Nie mogę znaleźć żadnego błędu w wierszu wskazanym przez kompilator. Co mogę zrobić?

O. Numer wiersza wyświetlany w komunikacie o błędzie nie zawsze oznacza miejsce, w którym należy usunąć błąd. Przyjrzyj się instrukcjom znajdującym się bezpośrednio nad komunikatem o błędzie i sprawdź, czy nie zawierają one literówek lub innych błędów. Błąd z reguły znajduje się w tym samym bloku programu.

P. W jaki sposób mogę odwiedzić Antarktydę?

O. Jeśli nie zamierzasz zostać badaczem naukowym lub pracownikiem obsługi (np. kucharzem, elektrykiem lub lekarzem), możesz być jednym z 10 000 ludzi, którzy co roku turystycznie odwiedzają ten mroźny kontynent.

Wyloty są organizowane z Australii, Nowej Zelandii i Ameryki Południowej, a ich koszt wynosi około 1000 dolarów od osoby.

Kursują także statki rejsowe — takie wycieczki trwają od 10 dni do trzech tygodni, a najdroższe z nich kosztują około 25 000 dolarów. Niektóre z takich rejsów dają możliwość popływania kajakiem pomiędzy pingwinami, wejścia na górę lodową, a nawet przespania się w obozie.

Więcej informacji dla chcących zwiedzić Antarktydę znajduje się na stronie Polar Cruises (*www.polarcruises.com*).

Brytyjska organizacja Antarctic Survey ma dla takich turystów przydatną radę: „Nie wchodź na lodowce ani duże pola śniegu, jeśli nie masz odpowiedniego przygotowania".

Quiz

Poniżej znajdują się pytania, które pomogą Ci sprawdzić wiedzę nabytą w trakcie tej godziny.

1. Co tak naprawdę robisz, kompilując program?
   A) Zapisuję go na dysku.
   B) Konwertuję do postaci bardziej zrozumiałej dla komputera.
   C) Dodaję go do swojej kolekcji programów.
2. Czym jest zmienna?
   A) Czymś, co się kołysze, ale nie przewraca.
   B) Tekstowym elementem programu ignorowanym przez kompilator.
   C) Miejscem do przechowywania informacji podczas działania programu.
3. Jak nazywa się proces usuwania błędów?
   A) Demolowanie.
   B) Debugowanie.
   C) Dekomponowanie.

Odpowiedzi

1. **B.** Kompilowanie programu przekształca plik *.java* w plik *.class* lub zestaw plików *.class*.
2. **C.** Zmienne są jednym ze sposobów przechowywania informacji; w dalszej części książki dowiesz się o innych możliwościach, takich jak tablice lub stałe. Tym, co się kołysze, ale nie przewraca jest wańka-wstańka, a elementem programu ignorowanym przez kompilator jest komentarz.

3. **B.** Ponieważ błędy w programie komputerowym są nazywane bugami, proces usuwania tych błędów nazywa się debugowaniem. Niektóre narzędzia programistyczne są wyposażone w debugger, który pomaga eliminować błędy. NetBeans ma jeden z najlepszych debuggerów.

Ćwiczenia

Jeśli chcesz trochę dokładniej zgłębić tematy opisane w trakcie tej godziny, spróbuj wykonać poniższe ćwiczenia:

- Zdanie „Witaj, świecie!" możesz przetłumaczyć na inne języki, używając Tłumacza Google dostępnego pod adresem *http://translate.google.com*. Napisz program, za pomocą którego komputer przywita się ze światem, np. w języku francuskim, włoskim lub portugalskim.
- Wróć do programu Saluton i wprowadź do niego jeden lub dwa błędy. Przykładowo, usuń średnik z końca wiersza lub zmień tekst `println` na `println` (wprowadzając cyfrę 1 zamiast litery `l`). Zapisz program i spróbuj go skompilować, a następnie porównaj komunikaty o błędach z wprowadzonymi błędami.

Rozwiązania ćwiczeń znajdziesz na serwerze FTP wydawnictwa: *ftp://ftp.helion.pl/przyklady/jav24h.zip*.

Skorowidz

E

F

G

H

I

J

K

L

Ł

M

N

O

P

R

S

Ś

T

U

W

Z

Ż

Java to język programowania, którego warto się nauczyć. To technologia nowoczesna i użyteczna, a jej możliwości doceniają największe firmy na całym świecie. Przy tym jest to język dojrzały i lubiany, wspierany przez rzeszę pasjonatów programowania. Na tym nie koniec zalet Javy: nauka tego języka jest prosta i przyjemna. Aby zacząć pisać aplikacje w Javie, wystarczy odrobina wysiłku i zaangażowania. Oznacza to, że jeśli zechcesz, wkrótce będziesz pisać programy z graficznym interfejsem użytkownika, łączące się z usługami sieciowymi, działające na urządzeniach mobilnych, a nawet kod pracujący w środowisku Minecrafta!

Ta książka to bardzo przystępny, zwięzły podręcznik składający się z 24 godzinnych lekcji. Ich celem jest nauka programowania w języku Java od podstaw — pojęcia programistyczne zatem są objaśniane w sposób zrozumiały, a technika tworzenia kodu została opisana krok po kroku. Po 24 godzinach spędzonych z tą publikacją będziesz pisać własne programy. Nauczysz się projektowania graficznych interfejsów użytkownika, tworzenia aplikacji mobilnych, zrozumiesz zasady programowania obiektowego. Ta książka przygotuje Cię do nauki bardziej zaawansowanych technik programowania w Javie.

Najważniejsze zagadnienia:

- **Konfiguracja środowiska programistycznego Javy**
- **Podstawowe elementy kodu**
- **Budowanie funkcjonalnych interfejsów użytkownika**
- **Stosowanie wątków i praca z plikami**
- **Techniki programowania obiektowego**
- **Tworzenie aplikacji mobilnych**

Rogers Cadenhead jest programistą, twórcą aplikacji, autorem publikacji i wydawcą internetowym. Napisał już ponad 20 książek o tematyce informatycznej. Przez kilka lat był członkiem grupy RSS Advisory Board, która zajmuje się specyfikacją RSS. Swoje artykuły publikuje w Drudge Retort i w innych popularnych serwisach internetowych.

Programowanie w Javie?
Łatwiejsze, niż się wydaje!

SAMS

Helion

helion.pl

HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-4235-4

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 69,00 zł